WHAT IS CLAIMED IS:

1. A method for transferring data, comprising:

providing a socket to create a virtual connection to transfer data between a first application process residing on a computer and a computer network; and

releasing the socket when the data has finished transferring to allow the virtual connection to transfer data between a second application process and the computer network.

2. The method as set forth in claim 1, further comprising providing a plurality of sockets to create additional virtual connections between application processes and the computer network.

3. The method as set forth in claim 2, further comprising assigning each of the application processes to an available one of the plurality of sockets.

4. The method as set forth in claim 3, wherein each of the plurality of sockets is determined not to be in use prior to the assigning.

5. The method as set forth in claim 3, wherein assigning is performed using at least one of the following assignment techniques: (a) round robin; (b) random; (c) user defined.

6. A computer-readable medium having computer-executable instructions for performing the method as set forth in claim 1.

7. A method for transferring data between a computer having a processor and a computer network, comprising:

creating a plurality of sockets capable of providing virtual connections between processes executing on the processor and the computer network; and

assigning each of the processes to an available socket of the plurality of sockets in response to a request for a socket.

8. The method as set forth in claim 7, wherein assigning is performed using a round robin socket assignment technique.

9. The method as set forth in claim 7, wherein assigning is performed using a random socket assignment technique.

10. The method as set forth in claim 7, wherein assigning is performed using a user-defined assignment technique.

11. The method as set forth in claim 7, wherein the data is divided into separate data units.

12. The method as set forth in claim 7, wherein the data are incoming network requests and the incoming network requests are demultiplexed into separate network requests corresponding to a data unit.

13. A data transfer system for transferring network data, comprising:

a plurality of sockets for providing a virtual connection between a computer and a computer network;

a plurality of threads for processing the network data, each one of the plurality of threads capable of being assigned one of the plurality of sockets; and

a parallel sockets module in communication with the plurality of sockets and the plurality of threads that provides parallel transfer of the network data using the plurality of sockets.

14. The data transfer system as set forth in claim 13, wherein the parallel sockets module further comprises a network data processor that divides the network data into a plurality of data units.

15. The data transfer system as set forth in claim 13, wherein the parallel sockets module further comprises an assignment module that uses a socket assignment technique to assign at least one of plurality of threads to an available one of the plurality of sockets.

16. The data transfer system as set forth in claim 15, wherein the parallel sockets module further comprises a binding module that binds the assigned thread to the assigned socket.

17. The data transfer system as set forth in claim 15, wherein the socket assignment technique is a round robin technique that assigns the thread to a first available socket.

18. The data transfer system as set forth in claim 15, wherein the socket assignment technique is a random technique that assigns the thread randomly to an available one of the plurality of sockets.

19. The data transfer system as set forth in claim 15, wherein the socket assignment technique is a user-defined technique that assigns the thread to an available one of the plurality of sockets as determined by a user.

20. A computer-implemented method for transferring data between a network server and a computer network, comprising:

providing a socket having a virtual connection between first server process on the network server and the computer network;

determining that the socket is available;

binding the first server process to the available socket to facilitate the transfer of data; and

making the socket available to a second server process when the data has finished transferring.